GENCAD: A HYBRID ANALOGICAL/EVOLUTIONARY MODEL OF CREATIVE DESIGN

ANDRÉS GÓMEZ DE SILVA GARZA
Instituto Tecnológico Autónomo de México (ITAM)
Río Hondo #1, Colonia Tizapán San Ángel
01000—México, D.F.
México

AND

MARY LOU MAHER
Key Centre of Design Computing and Cognition
Department of Architectural and Design Science
University of Sydney NSW 2006
Australia

Abstract. Computational models of creative design typically adopt a specific mechanism and apply that mechanism to a design space to demonstrate its potential for generating creative designs. For example, some computational models use analogy as a reasoning mechanism for developing novel designs from previous designs and others use evolutionary algorithms as a way of searching the design space to get to parts of the space that may not be visited using other search techniques. In this paper we present a computational model of creative design that combines analogy and evolution in an integrated framework, drawing on the strengths of each for generating creative designs. We describe the process model, present and discuss a sample application domain, and give some experimental results. In the end, we evaluate the combined computational model in terms of the reasoning process followed by the model and the designs that the model generates.

1. Introduction

In previous Heron Island conferences different process models have been proposed for creative design. Some of these computational models have used analogy or case-based reasoning as their basis for generating designs (e.g., (Prabhakar and Goel, 1992), (Qian and Gero, 1995), (Do and Gross, 1995), (Wolverton and Hayes-Roth, 1995), (Gero and Kazakov, 1998), and (Gomes et. al, 1998)). Other process models have focused on using concepts from evolutionary algorithms for the creation of new designs (e.g., (Gero, 1992), (Lenart and Pasztor, 1992), (Louis, 1992), (Graf and Banzhof, 1995), (Gero and Kazakov, 1998), (Parmee and Bonham, 1998), (Jo, 1998), and (Maher and Wu, 1998)). Zhao and Maher (1992) presented two computational models of creative design, the first based on analogy and the second based on mutation, one of the operators used in evolutionary algorithms. Analogy and evolution have been applied together in different ways to model creative design (for example, (Louis et. al, 1993), (Ramsey and Grefenstette, 1993), and (Gero and Schnier, 1995)). In contrast, we present a single process model of creative design that integrates analogy and evolution.

Our model, GENCAD, is a process model of creative design that integrates analogy and evolution into a single reasoning framework. A similar combination is presented in (Rosenman, 2000), which uses a genetic algorithm as a mechanism for the case adaptation process. The emphasis in our model is the potential for this approach to model creative design.

Using analogy for problem solving implies that new problems are solved based on the solutions to previously known problems that are in some way similar or relevant to the new problem. This means that previously encountered problems (and their associated solutions, and any other relevant or important information about them) have to be stored in a case base in a system’s memory. Even a small case base can require a lot of programming to be done before the computational system can begin to be used. Normally, reusing a solution to an old problem to solve a new problem cannot be done directly—the old solution will have to be adapted in some way by the system to fit the context of the new problem before it can be used to generate a solution for the new problem. This case adaptation task usually needs even more knowledge (e.g., in the form of adaptation heuristics or rules obtained from human experts or other sources) to be programmed into the system. Both the cases themselves and the case adaptation knowledge result from a lengthy, iterative, and often imprecise or inconsistent knowledge acquisition process. Both the cases and the case adaptation knowledge will differ from one application domain to another and from one type of reasoning task to another, so they usually can’t be reused if the application domain changes. Analogy is thus a heavily knowledge-intensive problem-solving method.

Using evolution for problem solving implies that solutions to new problems are slowly evolved out of randomly-generated initial solutions. At each evolutionary step, genetic operators such as combination and mutation are applied at random to a population of proposed solutions, resulting in the generation of new possible solutions. The new solutions are evaluated by applying an evaluation function to them, resulting in a fitness value being assigned to them. The best of the proposed solutions (out of the set of old and new ones) are kept after each evolutionary step, and the worst ones discarded; this is done by comparing their relative fitness values. The average quality of the solutions in the evolutionary algorithm's population thus improves over time. The process is repeated cyclically through several evolutionary generations until one or more solutions have been found that are of satisfactory quality (depending on how the fitness value is measured, assigned, and interpreted). Evolution does not need any domain- and task-specific knowledge in order to generate new solutions, since its operators do not need to know the semantics of the "genetic material" (the proposed solutions in the population) that they operate on. The only subtask of evolutionary algorithms in which knowledge is needed is in the evaluation of the proposed solutions. Evolution is thus a knowledge-lean problem-solving method.

Analogy and case-based reasoning have been widely and successfully used in Artificial Intelligence for solving many different types of problems (see (Kolodner, 1993) or (Leake, 1996) for many examples), including design tasks. Evolutionary algorithms have also been widely used in Artificial Intelligence for solving many different types of problems (see (Mitchell, 1998) for many examples), including design tasks. The different nature of these two types of problem solving thus does not influence their effectiveness. Analogy requires a laborious and often problematic knowledge acquisition phase, but the resulting expert knowledge guides the process quickly towards satisfactory solutions; it does not rely on making some decisions at random and hoping for the best. Evolution does not require much knowledge acquisition, but can effectively search larger spaces, often exploring areas which ultimately prove fruitless, in order to find satisfactory solutions; it does not use any *a priori* knowledge to guide its search. The strength of one of these two problem-solving methods is the weakness of the other, and *vice versa*. But used jointly, analogy and evolution can complement each other and result in a more robust problem-solving method than either method can be by itself.

In section 2 we present GENCAD (GENetic Case ADaptation), our process model for hybrid analogical/evolutionary creative design. In section 3 we discuss a sample application domain, residential layout design. In section 4 we present our application of the GENCAD process model, which uses Frank Loyd Wright prairie houses as a case base for analogical reasoning and the *feng shui* domain as the fitness function for the genetic algorithm. In section 5 we evaluate GENCAD's creativity according to several process-centred and product-centred criteria. Section 6 closes the paper by summarising and discussing the material presented in the other sections.

2. GENCAD: An Analogical/Evolutionary Model of Creative Design

GENCAD's process model emphasises solving problems using information derived from precedents; this is the main idea behind case-based reasoning or analogy. The solutions to past design problems contained in the precedents that are retrieved from memory serve as starting points. Multiple random combinations and modifications (together referred to as adaptations) of the retrieved cases are then generated and evolved incrementally, until a satisfactory solution to the new design problem is found; this is the main idea behind evolutionary algorithms. Figure 1 shows GENCAD's process model for case-based creative design with evolutionary design case adaptation.

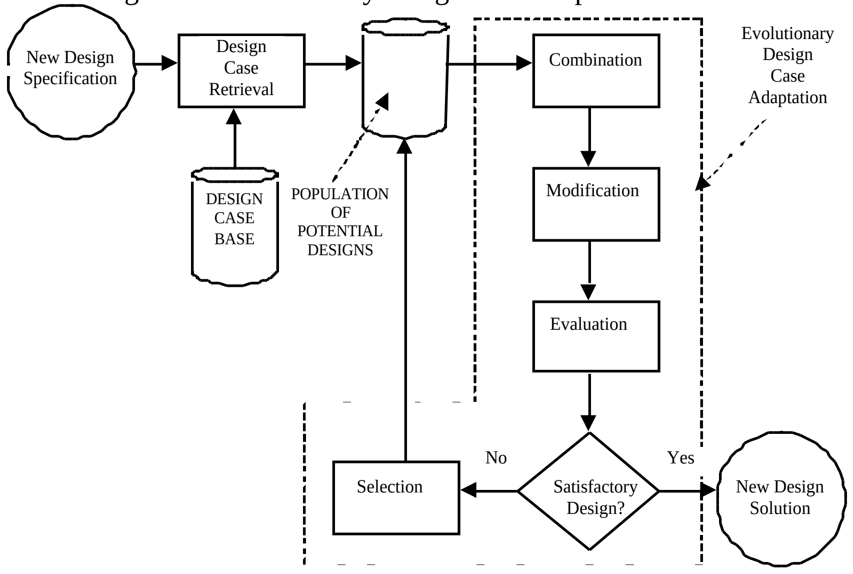


Figure 1. A process model for case-based design with evolutionary case adaptation.

2.1. OVERVIEW OF THE PROCESS MODEL

Given the specification of the requirements of a new design problem, the first task in the process model is to determine which precedents contain information that might be useful in solving the new problem. This is done by consulting a case base and comparing the description of the new problem with the descriptions of the precedents stored in the case base. Those cases for which some similarity is found with the new problem are retrieved from the case base. They are taken to be first approximations towards a solution to the new problem, and are put together into a population of potential designs. The cases in this population need to be adapted to become satisfactory solutions to the new problem.

This task of case adaptation is performed by an evolutionary method. Two types of adaptation are available to the evolutionary method: combination and modification. These types of adaptation are performed on the designs in the population through the genetic operators of crossover and mutation, respectively. Both crossover and mutation involve random decisions; thus, the precise appearance of the new designs resulting from adapting the old designs cannot be predicted before performing the adaptations. Crossover produces two offspring designs, each of which combines features from each of two parent designs. Mutation produces one offspring design which is an altered version of one parent design. Both crossover and mutation insert new designs into the population.

The two types of adaptation can result in generating offspring designs that are better (i.e., closer to being a solution to the new design problem) than some or all of the old parent designs, or worse. In order to determine this, the potential solutions have to be evaluated and their relative worth compared. In the process of evaluating the solutions, one (or more) might be found that are good enough to satisfy the requirements of the new design problem. If this happens, the process model can stop at this point; if not, the evolutionary adaptation process continues, using the best of the designs as the initial population for the next cycle of adaptations. The best designs are selected from the augmented population containing both old (relative to the current evolutionary cycle) and newly generated potential designs.

Evaluation and selection are not directly responsible for manipulating old designs to produce new ones, unlike combination and modification. However, they are considered part of the evolutionary case adaptation process because the results they give define the paths examined during the search for a solution to the new problem. Evaluation and selection also define which designs will be available for adaptation in future cycles; thus, they guide and influence case adaptation.

More detailed explanations about the subtasks in GENCAD’s process model can be found in (Gómez de Silva Garza, 2000) and (Gómez de Silva Garza and Maher, 2000). The following section discusses some issues related to the representation of the knowledge used by GENCAD’s process model which have an impact on the process model.

2.2. KNOWLEDGE REPRESENTATION ISSUES

In this section we present some issues related to knowledge representation and the semantics of the knowledge used by the process model that have a bearing on the implementation of the processing aspects of the model. We consider here the contents of each represented case and the evolutionary algorithm notions of phenotype and genotype (and how the concept of a case relates to them).

2.2.1. Contents of a Case

An important consideration when working on a case-based reasoning (CBR) project is to decide on the types of information that will be held in the cases in memory. Depending on the CBR system's task, application domain, and the role that cases play in the system, a specific case representation is decided upon. Usually, each case within a particular CBR system's memory will contain the same types of information as every other case in the same case memory. In general, every case in every CBR project contains a description of the solution to a previously encountered problem represented in one way or another. But in some projects the cases in addition might store a description of:

- the problem that was solved by the solution,
- the problem-solving steps that were taken to reach the solution,
- annotations, explanations or justifications of aspects of the solution, and/or
- annotations, explanations or justifications of the problem-solving steps that were taken in generating the solution.

In addition, if CBR is being used for design, even the description of the solution to a previously encountered problem can include not just a description of the designed artefact, but also some contextual or support information. For instance, the description of the solution might also include a description of the environment in which the designed artefact is meant to operate in or other related information that might be useful when reusing the design and that might influence some design decisions.

Not all such knowledge that can be stored together with a case might be needed in the case adaptation process. For instance, it could be that only the description of the solution itself, without any added contextual information, is required during case adaptation (while the supporting information may have been useful for performing case retrieval or for communicating the contents of the cases to the user in different ways).

Implicit in GENCAD's process model, therefore, is the possibility that some preparation will be required of the cases retrieved from memory before they can become the initial population of the evolutionary case adaptation method. Figure 2 shows this expanded view of the case retrieval task from the process model.

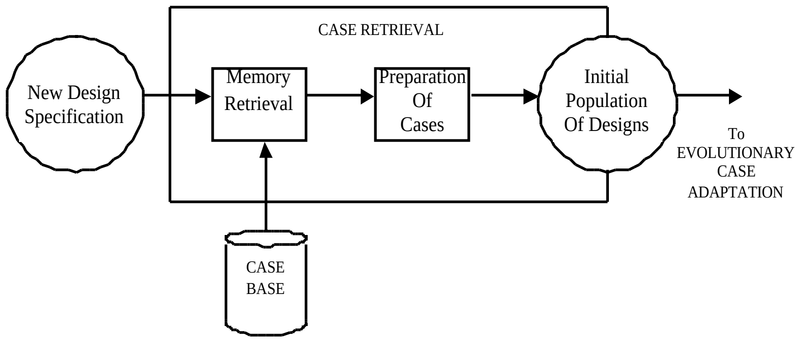


Figure 2. An expanded view of case retrieval.

The extra preparation phase, for instance, might require stripping away irrelevant information, such as information about the context that is fixed in the new design description and therefore should not be adapted, or re-representing some of the information stored in the case in some way that is meaningful to the case adaptation task. Since our process model is general and makes no commitments about the exact content or representation of cases, this potential preparation phase has to be considered separately for each individual application of the model. In subsequent discussions of the evolutionary part of the process model, if reference is made to the cases in the population, it will be to the entities resulting from this preparation phase, if any (in other words, the entities that become the initial population of the evolutionary method).

2.2.2. Phenotype-Genotype Dichotomy in Evolutionary Algorithms and Its Relation to Cases

In evolutionary algorithms a distinction is made between the phenotype and the genotype of an individual in an evolving population. The genotype is the genetic material that describes the individual using a set of primitive genetic instructions/units. The phenotype is the actual physical, behavioural, and functional characteristics displayed by the individual after the genetic instructions in the genotype are interpreted. The synthesis-related subtasks of an evolutionary algorithm, crossover and mutation, operate on genotypes: it is genetic material that gets transformed by these operations. The analytical subtasks of an evolutionary algorithm, evaluation and selection, operate on phenotypes. This is because it is the characteristics of the individual resulting from the genetic changes that are important in deciding whether the individual is good enough to survive in future generations (or to be a solution to the problem currently being solved).

The implications of the phenotype-genotype dichotomy for GENCAD's process model are as follows. First, though conceptually a phenotype is different from a genotype, in practice for a given evolutionary algorithm it might be decided that it is not necessary to encode or re-represent phenotypic characteristics within an equivalent genotype. In this case a population of phenotypes and one of genotypes might to an outside observer look the same, since they will both consist of entities that follow the same representation scheme, are of the same size, and generally cannot be distinguished from each other. However, we will keep the dual phenotype-genotype terminology, both in order to maintain conceptual clarity and because we want the process model to be as general as possible.

In many situations it might be decided that different representations are required or desirable for phenotypic and genotypic characteristics. In this case, a population of phenotypes will have to be transformed into a population of equivalent genotypes before crossover and mutation can be performed. The new population of genotypes produced after crossover and mutation will have to be transformed into a population of phenotypes before evaluation and selection can be performed. Thus, implicit in the process model are two potential mapping phases of the evolutionary method, a phenotype-to-genotype transformation before crossover and mutation, and a genotype-to-phenotype transformation before evaluation and selection. Each phenotype-genotype pairing can be stored, for instance using an associative memory, once a genotype-to-phenotype or phenotype-to-genotype transformation has been performed (we are assuming here that there is a one-to-one mapping between genotypes and phenotypes—otherwise things can get more complicated). If this is done, then subsequent "transformation" steps involving the same individual can be performed by consulting the associative memory pairing equivalent phenotypes and genotypes, rather than by performing the transformation from scratch.

Figure 3 shows this expanded view of the evolutionary case adaptation task in GENCAD's process model.

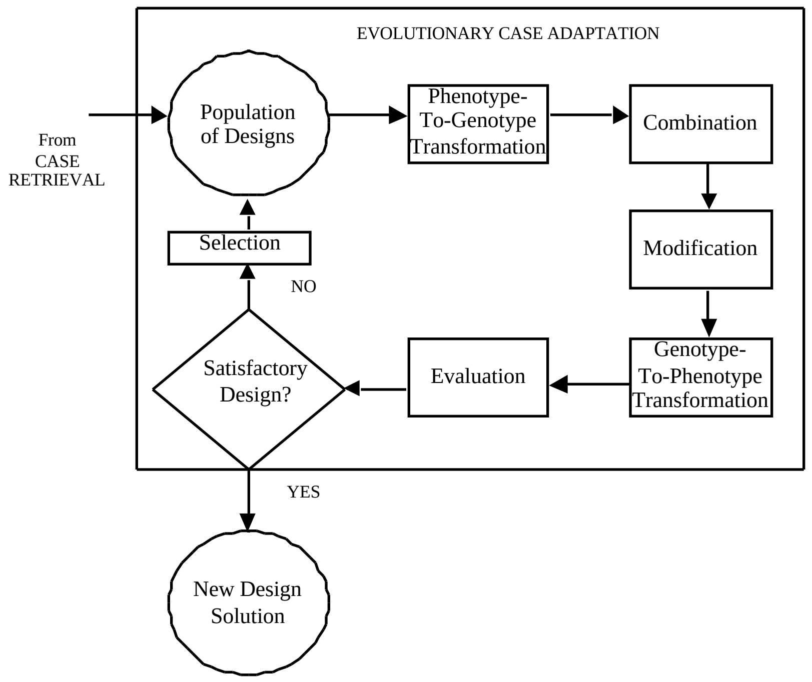


Figure 3. An expanded view of evolutionary case adaptation.

The cases forming the initial population of the evolutionary method for design case adaptation may contain descriptions of the functional, behavioural, and/or structural aspects of previously known solutions (or whatever other breakdown of the information, if any, is chosen). Thus, they are phenotypes. However, even though all cases are phenotypes, not all phenotypes are cases. When new genotypes are created through crossover and mutation, the results are then re-interpreted as phenotypes. These phenotypes are not cases, because they are not previously encountered solutions, i.e., they are not precedents or experiences, and they do not reside in long-term memory.

To sum up, we have presented a process model for the evolutionary adaptation of design cases. This process model combines aspects of CBR and evolutionary algorithms into one framework for adaptive design. The four subtasks of the evolutionary algorithm, combination, modification, evaluation, and selection, together guide the search for a solution to a new problem, while at the same time the search has a degree of flexibility due to the random nature of some of the decisions made in the subtasks. The search is given additional initial bias by the use of cases. Cases are retrieved from a case memory based on their relevance to the requirements of the new problem and are used to initialise the population of potential solutions of the evolutionary algorithm. In the process model, knowledge is only required in the form of cases which provide a framework for the construction of new potential solutions, and in the form of domain constraints which help evaluate potential solutions generated by the evolutionary algorithm.

3. *Feng Shui* Residential Layout Design

In this section we discuss the application domain of design of residential layouts that satisfy the principles of *feng shui*. We also present some knowledge representation issues introduced by this application domain.

3.1. CHARACTERISTICS OF *FENG SHUI*

Feng shui, also known as Chinese geomancy, is an ancient Chinese technique that, among other things, determines the quality of proposed or existing layouts of residences according to several rules of thumb. Some of these heuristics seem to have a basis in common sense, or in a psychological or sociological appreciation of the human beings that inhabit (or intend to inhabit) the residence. Other heuristics seem to be of a more superstitious nature.

There are several different *feng shui* sects that may contradict each other or place different priorities on different aspects of residential layouts. Some *feng shui* heuristics focus on the date of birth, profession, gender, income, and family relationships of the people that the residence is or will be for. They might interpret the quality of a design according to a combination of these social characteristics with some geographical features such as absolute cardinal directions (north, south, etc.). Other *feng shui* heuristics might emphasise more the presumed speed at which *ch'i* (positive energy assumed to emanate from living creatures and animate objects) is allowed to circulate or flow.

Despite this variety, of prime importance to performing a *feng shui* analysis according to the principles of any of the existing sects is information on the relative positions of objects. In addition to their positions, other attributes of objects are usually also taken into account, such as their orientations, shapes, and relative sizes. In this paper we have used the knowledge of *feng shui* presented in (Rossbach, 1987), which corresponds to the Tibetan black-hat sect of *feng shui*.

Feng shui analyses different aspects of a residential layout to determine its auspiciousness or lack thereof. Some classes of inauspicious layouts can be “cured” by the proper placement of an acceptable curing object. Thus, some potentially bad layouts can actually be acceptable if the proper cure is present. Given a layout, it is not just a matter of determining whether the layout is “good” or whether something about it is “bad,” because even if some aspect of it makes it bad, one has to determine whether a valid cure for its bad features is present or not before rejecting the layout outright.

The *feng shui* knowledge contained in (Rossbach, 1987) applies to three different levels of description of a residence:

· the landscape level (the location of a residence with respect to other objects in its environment such as mountains, rivers, roads, etc.),

· the house level (the relative placement of the rooms and functional spaces within a residence, such as bedrooms and bathrooms, as well as the connections between them, such as doors and windows), and

· the room level (the location of furniture, decorations, and other objects within each room or functional space in a residence).

Feng shui analysis assumes knowledge of spatial relationships among the objects at each of the different levels. Absolute locations and exact measures of distances and other geometric quantities are not as important. Because of this, a qualitative spatial representation of location seems to be sufficient.

3.2. SPATIAL REPRESENTATION FOR *FENG SHUI*

The representational framework we use to describe the locations of objects within each of the three levels is a low-resolution, qualitative one. We locate objects (which we will refer to as elements) at each level on a 3x3 spatial grid, with each sector within the grid assigned a unique number between 1 and 9 to identify it. The grid is shown in Figure 4, with north assumed to be at the top of the grid.

1	2	3
4	5	6
7	8	9

Figure 4. Qualitative spatial grid for elements.

Elements can occupy more than one grid sector, and grid sectors can contain more than one element, making the representation flexible. The resolution of this representation is not high, but considering the qualitative nature of a typical *feng shui* analysis and the number of elements that typically need to be represented at each of the three levels, it is adequate in most cases. It allows us to determine the relative positions of objects, which is the most important capability needed for an adequate *feng shui* analysis.

Figure 5 shows a typical residence viewed at the three levels of description that a *feng shui* analysis would look at, with the 3x3 spatial grid superimposed on each level. The figure is immediately followed by an equivalent symbolic representation of the residence shown in it. This demonstrates how the relevant information can be represented in the computer.

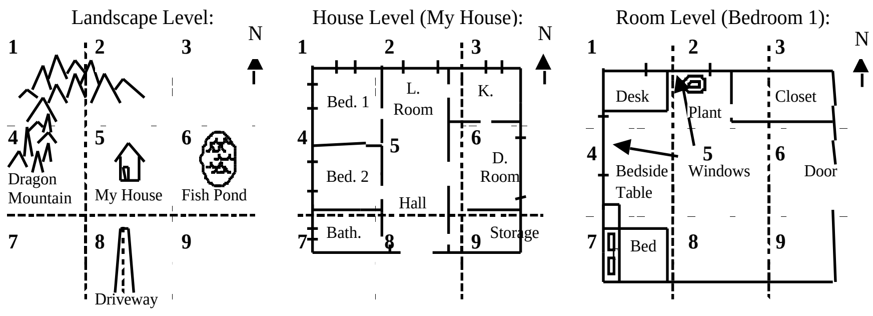


Figure 5. Example of a residence viewed at the three levels.

```
(((level landscape)
  (elements (((type hill) (name dragon-mountain)
              (location (1 2 4)) (steepness high) ...)
             ((type house) (name my-house) (location (5)))
             ...)))
 ((level house)
  (elements (((type bedroom) (name bedroom-1) (location (1))
              (shape square))
             ((type bedroom) (name bedroom-2) (location (4)))
             ((type hallway) (name hall-1) (location (5 8)))
             ...))
  (connectors (((type internal-door) (name b2-hall) (location (5))
                (side-a bedroom-2) (side-b hall-1)
                (direction ew))
               ((type window) (name b1-window-1) (location (1))
                (side-a bedroom-1) (side-b outside)
                (direction ew))
               ...)))
 ((level room)
  (name bed-1)
  (elements (((type bed) (name b-1) (location (7)))
             ((type desk) (name d-1) (location (1)))
             ((type window) (name w-1) (location (1 2)))
             ...))))
```

4. Implementation and Experiments

In this section we present our implementation of the GENCAD process model to the *feng shui* domain, called GENCAD-FS. In this presentation we only consider the house level of description of residences, since it offers more implementation challenges and representational intricacies than the landscape or room levels of description. After GENCAD-FS is discussed, an experiment and its results are described.

4.1. REPRESENTING DESIGN PROBLEMS IN GENCAD-FS

A person who is about to design a residence might specify, for example, that it must have three bedrooms, two bathrooms, and a music room, and that in addition it must satisfy the principles of *feng shui*. The design task then becomes to decide how to place and connect the specified spaces in a configuration that is acceptable according to *feng shui*. Our process model makes no commitments about how design problems can be specified and represented. However, one possible representation for the requirements of a new problem, which we used in GENCAD-FS for design problems such as the one outlined above, is as a list of structural element types associated with a quantity, where this quantity represents the number of structural elements of the given type that are required. For instance:

```
((bedroom 3) (bathroom 2) (music-room 1))
```

Choosing this representation for the problem requirements has implications for the exact way in which the case retrieval task is implemented (i.e., on how matching of these requirements with the information in the cases in memory is done). It also influences the way in which the evaluation module must determine the problem fitness of a proposed design (i.e., the part of its fitness related to how good the design is according to the requirements of the current problem, rather than according to any constraints imposed by the domain). Whatever the representation framework chosen for design problems is, the process model remains unchanged.

However case retrieval is done, it still occurs, and the retrieved cases still become the initial population of an evolutionary algorithm that performs case adaptation. Matching a proposed design with the requirements of the current problem in the evaluation module to determine the design's problem fitness is done in an analogous fashion to matching a case in memory with the problem requirements during case retrieval. Again, the exact way in which this is done differs depending on the way in which problem requirements are specified and represented, but the process model remains unchanged.

4.2. DESIGN CASES AND CASE MEMORY IN GENCAD-FS

The *feng shui* design of residences is a design domain that has not been formalised to the extent that other design domains have. *Feng shui* practice has been passed on through the ages by word of mouth and simple sketches, rather than as a scientific body of published data and guidelines. Even today there are very few textbooks on the subject, and those that exist present the subject matter in a rather informal way. In addition, there are multiple sects of *feng shui* which often contradict each other in their interpretation of what is important, how to create designs that are acceptable, and/or how to "cure" bad designs to make them acceptable.

Because of this, it would be very difficult to find collections of floor plans of residences that conform to (a given sect of) *feng shui*. Such libraries of precedents just do not exist. In theory it would be possible to compile listings and floor plans of existing residences that follow a given *feng shui* sect, by consulting practitioners that have been involved in the design of such residences. However, this laborious step would have to precede the process of capturing the descriptions of the cases in the computer, which itself is a complex and time-consuming knowledge engineering task.

However, our process model does not specify that both cases and domain constraints have to come from the same source or the same design culture. In fact, design cases describing floor plans that do not conform to the principles of *feng shui* can be used, and as long as the domain constraints used to evaluate possible solutions embody the required *feng shui* analysis knowledge, the solutions proposed by our process model will still be acceptable to *feng shui* practitioners.

In GENCAD-FS we used cases describing the floor plans of Frank Lloyd Wright prairie houses to form the case library. Some of these cases might not violate any of the constraints imposed by the *feng shui* application domain, but most probably would not satisfy all of the constraints. In any case, there is no reason to presuppose that it wouldn't be possible to find adaptations of these cases that would satisfy all of the *feng shui* constraints. The resulting designs might both satisfy the principles of *feng shui* and exhibit some of the stylistic characteristics of Frank Lloyd Wright's prairie houses.

In this way, the modularity of our process model enables us to use knowledge from different design cultures, styles, and communities in combination. We can use the process model to experiment with different combinations of design styles and cultures to produce interesting designs. We term this *inter-cultural design* or *cross-cultural design*. Using our process model for cross-cultural design can also be seen as a way of "correcting," "improving," or "redesigning" cases according to design criteria that are different from those employed when the cases were originally designed.

GENCAD-FS's case library currently contains 12 cases, each of which describes one of Frank Lloyd Wright's prairie houses. These cases were obtained from (Hildebrand, 1991). The cases that have been implemented are: Hickox House, Bradley House, Willits House (lower floor), Heurtley House (lower floor), Heurtley House (upper floor), Cheney House, Avery Coonley House, Robie House (main floor), Robie House (upper floor), Hardy House (lower floor), Glasner House, and Roberts House.

It was decided to make all of the cases in GENCAD-FS's memory belong to the same style (in this case Frank Lloyd Wright's) in order to be able to analyse any solutions proposed by GENCAD-FS to determine whether they also appear to belong to the same style or not. This makes it easier to see that the solutions produced indeed contain features from one or more of the original cases in memory. It also provides us with an opportunity for combining in our implementation two very distinct design subcultures, *feng shui* analysis and Frank Lloyd Wright's prairie house design style.

A common theme in Wright's prairie houses is that they are usually designed to occupy two or even three stories, so many floor plans include staircases. One element that all cases have in common in GENCAD-FS is the presence of at least one fireplace, a standard element in Frank Lloyd Wright's prairie houses. Another typical Wright feature is the lack of "unnecessary" doors and partitions, i.e., the houses have many functional spaces that merge into each other without walls separating them.

Acquiring the house-level information for cases implied sometimes having to infer the functional use of a given space shown in the figures of floor plans presented in (Hildebrand, 1991). This was aided by the fact that most rooms and spaces in those figures are labeled, and most that are not labeled in one floor plan given appear equivalent both functionally and structurally to some that are labeled in a different floor plan. Thus, the inferences required were not many and are not likely to be misinterpretations of the floor plans.

At the house level, the cases in GENCAD-FS's case library represent a wide variety of designs. Some are second- or third-story floors, with many windows and terraces overlooking the landscape. Others are ground-level (or semi-subterranean) floors, which in Frank Lloyd Wright's designs often do not have many windows or are hidden from view from observers outside the house. Each floor of a multi-storied residence is represented as a separate case in GENCAD-FS. Thus some cases in the system's memory do not have a kitchen, or lack some other feature that would normally be present in a complete residence. This variety does not diminish GENCAD-FS's usefulness; in fact, if anything it increases its capacity for creativity.

4.3. REPRESENTING GENOTYPES IN GENCAD-FS

At the house level of the *feng shui* layout domain the important things are not just the elements that exist (such as bathrooms and bedrooms) and their features, but also the connections between elements (such as doors and windows) and from elements to the outside. Some of the features of the connectors, such as their orientation and type, are also important. A suitable genotype representation would have to include not just information about the existence and descriptions of elements, but also about their interconnections.

In order to capture this kind of information in GENCAD-FS we chose a genotype representation involving matrices, based on the concept of adjacency matrices (e.g., see (Chachra et. al, 1979)). A different type of matrix genotype, dealing with design elements at a much finer level of detail, has been used in (Kane & Schoenauer, 1996). Kane and Schoenauer also introduce different ways of performing crossover on such matrix genotypes. In this paper we will discuss only one possible matrix-based genotype representation that is useful at the house level of the *feng shui* domain, and one way of performing crossover on genotypes that are represented in this way.

Our matrix genotype expands on the concept of the adjacency matrix. Adjacency matrices help describe the connectivity between different nodes in a graph. A phenotype at the house level can be visualised using a graph representation in which the nodes can be the elements (rooms and functional spaces) and the arcs can be the connectors between elements (connections between rooms). Each node and each arc carries information associated with it describing the features of the corresponding element or connector. This type of phenotype can be converted into its equivalent adjacency matrix genotype before performing GA operations on it.

A basic adjacency matrix consists of K rows and K columns, where K is the number of nodes. The rows and columns of a given adjacency matrix are labeled in the same order, i.e., row I and column I both describe the same node. The item stored at location [I, J] in the matrix (where I<>J) is a number indicating how many arcs there are between node I and node J. This means that adjacency matrices are symmetric across their diagonal and have zeroes as the elements of their diagonals.

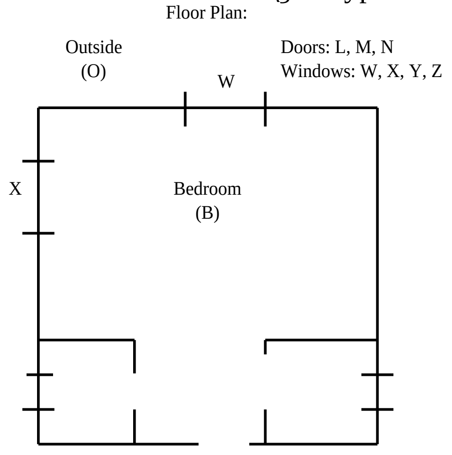
A matrix-based genotype expands on the concept of the adjacency matrix by:

· Including name vectors as part of the genotype. These name vectors label the rows and columns of the matrix. The name vectors contain the names of the rooms corresponding to each node. The names in the name vectors are used as pointers to the other attribute-value pairs describing the corresponding room when such additional information about a node has to be accessed.

· Including an extra node in the graph (and therefore an extra row/column in the matrix) to represent the outside (any space not considered to belong to the house).

· Including a list of which connectors connect node I and J at location [I, J] in the matrix (which indirectly gives us information about how many connectors there are), rather than just a number indicating how many connectors there are. The connectors named in the list point to the attribute-value pairs describing the connectors in more detail. Because a room or functional space cannot be connected to itself, each item on the diagonal of a matrix-based genotype will be the empty list.

Figure 6 shows a simple phenotype viewed as a floor plan (without superimposing the 3x3 grid on the plan, for simplicity), with its equivalent matrix-based genotype next to it:



$$\begin{array}{c|ccc} & B & & \\ \hline B & () & (& \\ T & (L) & & \\ K & (M) & & \\ & & & \end{array}$$

Figure 6. Graphical view of a phenotype and its equivalent matrix genotype.

Within a matrix genotype, the order in which genes appear does not matter (for example, B could have been placed in the second position instead of the first in Figure 6, and the genotype would have represented the same house layout), as long as the same order is followed in the rows and in the columns of the matrix. Phenotype-to-genotype and genotype-to-phenotype transformations at the house level involve converting from the symbolic representation to an equivalent matrix-based one, and *vice versa*.

4.4. CROSSOVER IN GENCAD-FS

At the house level, it is important to combine aspects from both the elements and the connectors of a residence so that each offspring genotype inherits characteristics of both the elements and the connectors of both parents. Thus, crossover should operate not just on the adjacency matrix, but also on the name vectors describing the elements that are used to label the rows and columns of the adjacency matrix. Performing crossover on two matrix-based genotypes produces two offspring genotypes that combine elements from each of the two houses, i.e., it achieves adaptation in the form of genotype combination. Crossover in GENCAD-FS implies the exchange of sub-matrices of the same dimensions. The diagonal of each sub-matrix has to correspond to part of the diagonal of the matrix genotype it is extracted from to ensure that the matrices resulting from crossover maintain their diagonal symmetry.

Crossover has been implemented in GENCAD-FS so that first the size of the sub-matrices that will be exchanged is randomly chosen (taking into account that their maximum possible size is the size of the smallest of the two parent matrices). Then, separately for each sub-matrix to be extracted, the index of its upper-left hand item is chosen at random relative to the parent matrix it is to be extracted from (taking into account the size of the sub-matrix, which places limits on the possible indices). Finally, the sub-matrix extraction and exchange is performed. In parallel with this operation, the same type of extract and exchange is performed on subparts of the name vectors associated to the parent matrices using the corresponding indices chosen at random for the sub-matrices.

Figure 7 illustrates what a typical crossover operation on a matrix genotype looks like in GENCAD-FS (including the vectors labeling the rows and columns of the matrices, which are considered to be part of the genotypes). The fragments of the genotypes that are cut and exchanged during crossover are shown enclosed in thick lines (which represent the crossover “points”). One of the parent genotypes is shown plain, while the other is darkly shaded. The figure shows how parts from both parent genotypes combine to create the offspring genotypes after crossover is performed at the indicated crossover points. The lightly shaded areas indicate parts of the offspring genotypes that were not exchanged during crossover, but that may have to be revised if the introduction of new adjacencies can lead to conflicts in some of the features encoded in the genotypes.

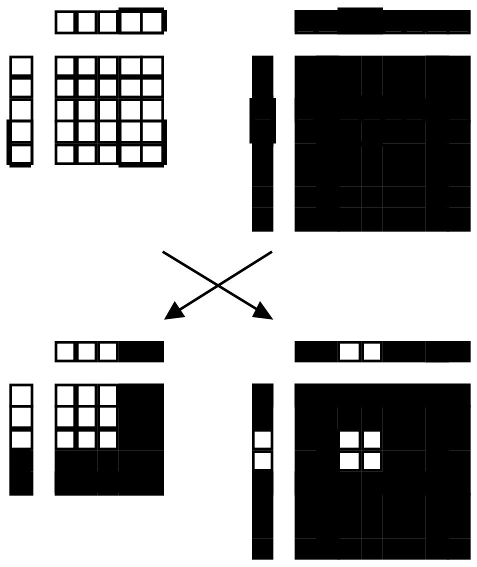


Figure 7. A typical crossover operation performed on two matrix genotypes.

The offspring resulting from such a crossover operation might introduce inconsistencies in the adjacencies to the rooms corresponding to some rows/columns of the parent genotypes, as shown in Figure 7, and it may be necessary to reconcile these inconsistencies. This reconciliation is performed in the implementation during the transformation of the offspring genotypes into their equivalent phenotype representations by modifying the value of the *location* attribute of some of the inconsistent rooms. If a full reconciliation of the locations of all of the rooms in an offspring genotype cannot be achieved, the offspring is discarded from the population (and more crossover operations are performed as needed to produce the required number of offspring at that evolutionary cycle).

4.5. MUTATION IN GENCAD-FS

Performing mutation at its most basic on a matrix genotype produces an offspring genotype that has the exact number of house-level elements of its parent, but one of the features of one of its elements and/or one of its connectors is changed. This achieves adaptation in the form of genotype modification. Figure 8 illustrates how a typical mutation operation in GENCAD-FS would modify a matrix genotype. The shaded squares indicate a gene that has been altered in the genotype during mutation. There are two of them because one of the triangles of an adjacency matrix is redundant (adjacency matrices are symmetric around their diagonal).

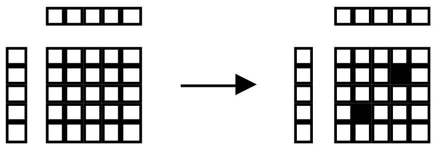


Figure 8. A typical mutation operation performed on a matrix genotype.

In GENCAD-FS we could have also proposed mega-mutation operators that alter entire elements or entire sets of elements or connectors of a genotype in one stroke (instead of just the value of one attribute of an element or connector at a time), but this would be like combining several mutations into one. Since mutation generally occurs, both in evolutionary algorithms and in nature, in small steps, we do not consider this possibility.

4.6. EVALUATION IN GENCAD-FS

The constraints used for evaluation in GENCAD-FS have been acquired from textual descriptions of *feng shui* principles presented in (Rossbach, 1987). Eight *feng shui* domain constraints and eight common-sense constraints have been implemented at the house level to determine an individual’s fitness. Common-sense constraints were included in order to filter out bad designs that don’t violate particular *feng shui* principles but would nonetheless not be feasible as proper designs of residences—for instance, designs in which there is a bedroom that is not connected to any of the other rooms in the residence.

Each constraint is implemented as a LISP procedure which returns a value of T if it is violated or a value of NIL if it is satisfied. In situations in which the *feng shui* principle associated with the constraint has a cure, verifying whether the cure is present or not is implemented as a separate

LISP function. The cure functions return T if a cure is present for the corresponding constraint, and NIL if no cure is present. Cure checking only occurs if the associated constraint appears to be violated, just before making the final decision on whether the constraint function should return T or NIL.
An example of a constraint at the house level is given by the following quote from (Rossbach, 1987):

> Traditionally, the Chinese avoid three or more doors or windows in a row...this...funnels ch'i [positive energy] too quickly...[CURE:]...to stop ch'i from flowing too quickly, hang a wind chime or crystal ball... (page 89)

This constraint is implemented by first finding the description of all the connectors at the house level, particularly their locations and directions. If at least three connectors are aligned such that their locations are in consecutive (or the same) grid sectors *and* they all have the same direction (e.g., north-south), then the constraint has been violated. However, before determining this we must check whether or not there are any wind chimes or crystal balls in the house that are positioned in line with the violating doors/windows. The pseudocode that represents this constraint procedurally is the following (given a phenotype P):

```
Get the list C of all connectors in P;
Get the list Q of all potential cures in P for this constr.;
For each connector c in C, or until a bad omen has been found:
   Get the location l of c;
   Get the direction d of c;
   Set the list of connectors LU lined up with c to NIL;
   Get the list Reduced of all elements in C except c;
   For each connector r in Reduced:
      If the direction of r is d And
         The location of r lines up with l along direction d,
         Then add r to LU;
      End-If;
   End-For;
   If there are two or more connectors in LU And
      no potential cure in Q lines up with r,
      Then signal a bad omen situation;
   End-If;
End-For;
```

Similar procedural representations have been implemented for the other constraints. A full list of the *feng shui* and common sense constraints that have been implemented in GENCAD-FS can be consulted in Appendix A of (Gómez de Silva Garza, 2000).

Some LISP predicates (functions that return T or NIL) were written to support the implementation of one or more of the *feng shui* domain constraints, as exemplified in the above pseudocode. Some of these support predicates are:

- *facing*: given the list of grid locations occupied by two objects and the orientation of the first, determines whether the first object faces (whether partially or completely) the second object or not,
- *behind*: given the list of grid locations occupied by two objects and the orientation of the second, determines whether the first object is located behind the second object (i.e., opposite the direction in which the second object is oriented) or not, and
- *contiguous-quadrants*: given two locations from the 3x3 grid used to represent spatial locations, determines whether they are adjacent or not.

4.7. SELECTION IN GENCAD-FS

Selection in GENCAD-FS consists of keeping the N best phenotypes in the augmented population (resulting from adding to an initial population the offspring generated through the crossover and mutation of the individuals in it), in order to use as the initial population for the next cycle of the evolutionary adaptation algorithm. The value of N is the number of cases that were initially retrieved from case memory; in this way the size of the population is kept constant across evolutionary cycles. These N best cases can include none, some, or all of the potential solutions generated during one evolutionary cycle, and none, some, or all of the potential solutions known at the beginning of the same evolutionary cycle. The exact makeup will vary greatly, depending on the relative quality of the newly generated offspring solutions, as determined during evaluation, and this will depend on whether the random decisions made during crossover and mutation are “lucky” (in that they result in improved solutions) or not.

4.8. EXPERIMENTAL RESULT

We have performed several experiments with GENCAD-FS. Complete descriptions of these experiments can be found in (Gómez de Silva Garza, 2000) or (Gómez de Silva Garza and Maher, 1999). Here we will only briefly describe one of the experiments. The following implementation strategies were followed when performing the experiment:

- The population size was kept constant across evolutionary cycles. If N cases were initially retrieved, then N is the size of the initial population for each generation of the GA.
- Within each evolutionary cycle, an intermediate augmented population was first obtained by adding to the initial population the offspring designs created through crossover. This intermediate augmented population was then used as a basis for performing mutation, and the designs created through mutation were added to the intermediate augmented population in order to create a final augmented population. This final augmented population was then culled by applying selection to it, thus creating the initial population of size N for the next evolutionary cycle.
- M new individuals were produced at each evolutionary cycle, of which 80% were created through crossover and 20% through mutation. We have set the value of M to equal the value of N (the number of initially retrieved cases).
- No biasing of the population according to the fitness values of the individuals in it (a process sometimes confusingly termed "selection" or "reproduction") was performed in our GA. All retrieved cases (or all phenotypes present in a population at any given time) may contribute a significant feature to the final solution, so all are treated equally.

The purpose of the experiment was to analyse the quality of the solutions found by GENCAD-FS by comparing their features to those stylistic characteristics normally present in the Frank Lloyd Wright cases that were used as a basis for constructing the solutions. For the experiment whose results we report here, the following house-level problem specifications were given:

```
((bedroom 3) (bathroom 2) (fireplace 1) (music-room 1))
```

That is, we asked GENCAD-FS to propose the floor plan of a residence with (at least) three bedrooms, two bathrooms, a fireplace, and a music room, such that their relative positions satisfy the principles of *feng shui*. The solution to this problem that was found by GENCAD-FS is shown in Figure 9.

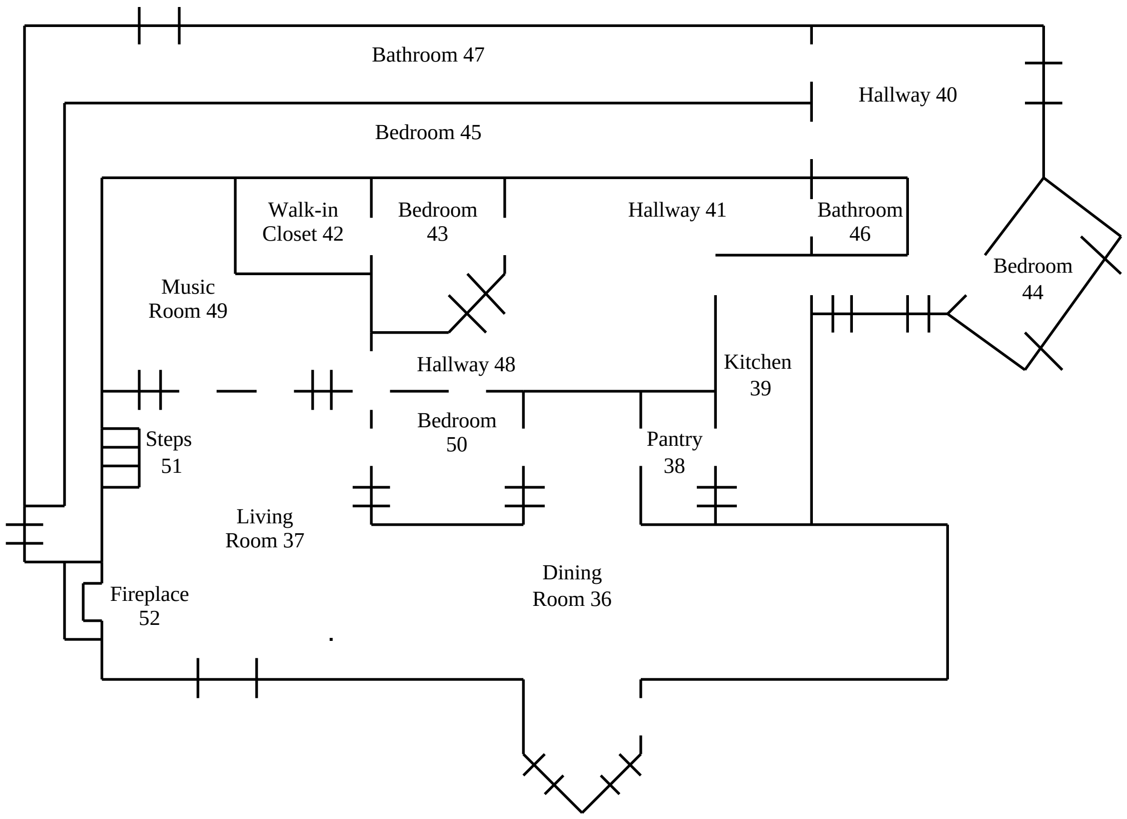


Figure 9. Solution found by GENCAD-FS.

We have implemented in GENCAD-FS some statistics-gathering code that allows us to trace the ancestry of each offspring design created in the process of solving a given problem. This allows us to make some observations about the way in which each design ever considered came to exist. Some characteristics of the process that GENCAD-FS underwent to produce the solution in Figure 9, as well as of the solution itself, are the following:

- All 12 Frank Lloyd Wright cases were retrieved from memory in order to participate in adaptation, as they all matched one or more of the problem requirements.
- GENCAD-FS converged after 61 evolutionary cycles to the solution shown.
- A total of 28 crossover operations and 3 mutations took place in order to generate this solution (not counting the crossover and

mutation operations whose results eventually caused a dead end during the search).

- The solution combines features inherited from three of the twelve cases that were originally retrieved: Avery Coonley House, Robie House (main floor), and Willits House (lower floor). Features from the other nine retrieved cases were probably still present in the population at the 61st generation, but not in the final solution (and were also never part of any of its ancestors).

5. Evaluating GENCAD's Creativity Potential

In order to evaluate GENCAD's creativity potential we can focus on two different aspects: its products (i.e., the designs it generates) and its process (i.e., the way it generates designs). In this section we will do both and show that GENCAD's process model can be considered creative according to several different criteria, both product- and process-centred. This evaluation could be applied to other computational models of creative design.

5.1. EVALUATING GENCAD ACCORDING TO PRODUCT

In our evaluation of the product of the design process, we are looking for recognisable features of previous designs combined in novel ways. The design solution shown in Figure 9 is an example of a product of GENCAD-FS, which is an instantiation of GENCAD. As such, any conclusions we make about the design shown in Figure 9 can be generalised to apply to GENCAD as a whole.

If we analyse the floor plan shown in Figure 9, and taking into account the observations on this solution made at the end of the last section, we can arrive at the following conclusions about the inheritance of individual features and the creativity of the resulting design:

- The fact that the solution is a floor plan that contains many hallways is a feature that is inherited from Avery Coonley House, which has 4 hallways. Neither of the other two ancestor cases has more than two hallways.
- The protuberance from the dining room is a feature that is inherited from Robie House, which has two such similar protuberances (one from the dining room and one from the living room), or from Willits House which has one (from the dining room).

· The room that is skewed compared to the north-south grid (Bedroom 44) is a feature that is inherited from Willits House, in which the rooms are all similarly skewed.
· The bedrooms in the solution are all in very different parts of the house, something that does not normally occur in a residence. In a family house, at least, the bedrooms are usually all near each other. However, if the residence were intended for several unrelated individuals sharing a house, the solution would be a creative and interesting design that would allow the occupants to have greater privacy than the design of a typical family house.

In other words, the solution combines (in a creative way) features from several known previous solutions. The solution also has some of the features considered to be general aspects of Frank Lloyd Wright's style, such as having large dining and living rooms that merge into each other, with no partitions between them, and having a fireplace. The fireplace was a problem requirement, so any solution had to have one to be a valid solution; its location in the living room was inherited from one of the three ancestor cases (all three had this feature). From the above, we can state that a product-centred evaluation of GENCAD allows us to conclude that it is a creative process model.

5.2. EVALUATING GENCAD ACCORDING TO PROCESS

When evaluating a process model's creativity according to process-centred criteria, one can focus on whether the process itself is creative (i.e., it is a creative way of solving problems that hasn't been tried before or extensively), or one can focus on the process's potential for producing creative products (whether or not the way the process operates is creative).

According to the first method, the GENCAD process model would be creative if analogy and evolution had not been used in this combination before. However, although it hasn't become too commonplace, there are design process models that combine ideas from analogy and evolution. Some examples are (Louis et. al, 1993), (Ramsey and Grefenstette, 1993), (Gero and Schnier, 1995), and (Rosenman, 2000). So, whether or not the GENCAD process model is creative is a subjective evaluation based on the degree of integration of these two disparate methods. Alternatively, we could draw on frameworks that characterise the creative potential of computational process models. Two such models are presented in (Maher et. al, 1995) and (Gómez de Silva Garza and Maher, 1998). In the next two subsections we will use these two frameworks to evaluate GENCAD's potential to generate creative designs. In both cases we will be able to conclude that GENCAD's process has a high potential for producing creative designs.

5.2.1. Exploration-Transformation Graph

In (Maher et. al, 1995), the authors consider that there is a zone of creativity potential that a process model can lie within. This zone of creativity potential forms part of the graph in which one plots the degree of exploration and the degree of transformation involved in the process model. Exploration is defined as operations that control the search within a design space, such as abstraction (a divergent operation, abstracting away from the details of an uninteresting area of the space) or refinement (a convergent operation, focusing the search in detail within a crucial part of the space). Transformation is defined as operations that alter the boundaries of a design space, such as adaptation (modification) and combination.

The degree of exploration and transformation of a process model in the (Maher et. al, 1995) framework is assigned one of three qualitative values: mundane, novel, and original. Mundane exploration is largely convergent. Novel exploration is both divergent and convergent. Original exploration is mostly divergent. Mundane transformation would be the result of tweaking the designs in the design space. Novel transformation would mean the modification of the range of values allowed for the variables involved in describing the designs in the space. Original transformation would imply altering the variables involved in describing the designs in the space. The zone of creativity potential is a diagonal region of the exploration-transformation graph in which neither exploration nor transformation are mundane, and in which both of them are not original (see Figure 10 below).

If we consider GENCAD, we can see that its exploration is mostly convergent, and thus is mundane: there is little abstraction or generalisation during GENCAD's search for a solution, but there is refinement through the adding of more and more details from more and more previously known solutions during its evolutionary search. On the other hand, GENCAD's process model involves original transformation: it performs both modification and combination (i.e., both parametric and structural adaptations) of the solutions in its design space, resulting in new designs that can be quite different from all of the designs originally there. These new designs can combine variables that did not exist in combination before in any of the known solutions in the design space, can involve new values assigned to these variables, and can propose new structures for the designs in the space. Figure 10 shows where GENCAD falls within the zone of creativity potential in the exploration-transformation graph.

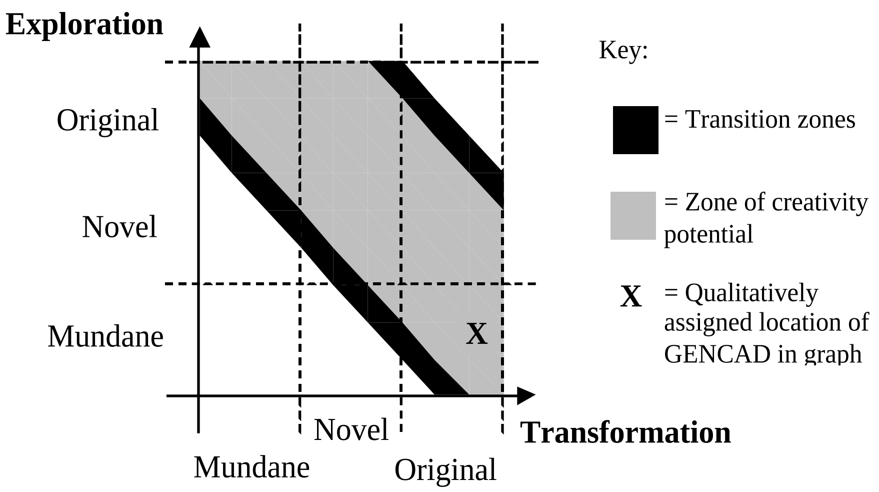


Figure 10. GENCAD's Place in the Zone of Creativity Potential.

5.2.2. Combination of Random Decision-Making and Knowledge

In (Gómez de Silva Garza and Maher, 1998), the authors conclude that the potential for creativity arises from a process model's use of both random decision-making and deep (i.e., domain- and task-specific) knowledge in order to solve a design problem. Neither process models that only use a lot of knowledge, but make no random choices, nor process models that only make a lot of random choices, but don't make use of any knowledge, can be considered as creative. In the first situation, the fact that no random decisions are made means that the process model's outcome can be predicted *a priori* (i.e., without even having to run the process model); if its products can be predicted, they can't really be considered creative. In the second situation, the fact that some spurious creative results can occur in a completely random "problem solver" cannot mean that the problem solver is creative, since most of the time the products it generates will be meaningless and thus worthless.

Thus, for a process model to have the potential for creativity, it must involve making some random decisions, but it must also involve the use of domain- and task-specific knowledge to guide it along its search for solutions. The use of random choice and the use of knowledge must be integrated in an "intelligent" way so that they complement each other. GENCAD satisfies these criteria. Knowledge in the form of design cases and domain constraints bias and guide GENCAD's search for a solution to a given problem. At the same time, random decisions in the crossover and mutation operators of GENCAD's evolutionary case adaptation algorithm mean that the products it generates cannot be predicted in advance. Because of this, GENCAD is a process model that has the necessary characteristics to produce creative designs.

6. Summary and Discussion

We have presented GENCAD, a process model for creative design that combines aspects of analogical (case-based) reasoning and evolutionary algorithms. From the point of view of case-based reasoning, the evolutionary algorithm in GENCAD performs the task of case adaptation. In order to do so it requires domain knowledge with which to recognise good solutions, but does not require domain knowledge with which to generate potential adaptations. This simplifies the implementation of the case adaptation task compared to "traditional" case-based reasoning systems. From the point of view of evolutionary algorithms, the cases retrieved by GENCAD given a new problem specification serve as an initial population of potential solutions for the algorithm to operate on. This gives the evolutionary algorithm an initial bias/help that "traditional" evolutionary algorithms, which are normally initialised with a population of randomly-generated potential solutions, don't have.

In order to evaluate GENCAD's creativity we have employed a systematic methodology, which we hope other researchers will adopt, based on analysing both the products it designs and the process it employs to generate those products. We have been able to conclude from *all* of the points of view used to evaluate GENCAD that it is indeed a process model with a high potential for creative design.

References

Chachra, V., Ghare, P.M., and Moore, J.M.: 1979, *Applications of Graph Theory Algorithms*, North Holland, New York.

Do, E. Y.-L. and Gross, M.D.: 1995, Drawing Analogies: Supporting Creative Architectural Design with Visual References, in J.S. Gero, M.L. Maher, and F. Sudweeks (eds), *Third International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, Australia, pp. 37-58.

Gero, J.S.: 1992, Creativity, Emergence, and Evolution in Design, in J.S. Gero and F. Sudweeks (eds), *Second International Round-Table Conference on Computational Models of Creative Design*, Department of Architectural and Design Science, University of Sydney, Australia, pp. 1-27.

Gero, J. and Kazakov, V.: 1998, Using Analogy to Extend the Behaviour State Space in Design, in J.S. Gero and M.L. Maher (eds), *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, Australia, pp. 113-143.

Gero, J. and Kazakov, V.: 1998, Adapting Evolutionary Computing for Exploration in Creative Design, in J.S. Gero and M.L. Maher (eds), *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, Australia, pp. 175-186.

Gero, J.S. and Schnier, T.: 1995, Evolving Representations of Design Cases and Their Use in Creative Design, in J.S. Gero, M.L. Maher, and F. Sudweeks (eds), *Third International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, Australia, pp. 343-368.

Gomes, P., Bento, C., and Gago, P.: 1998, A Function for Novelty Evaluation in Design, in J.S. Gero and M.L. Maher (eds), *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, Australia, pp. 145-160.

Gómez de Silva Garza, A.: 2000, *An Evolutionary Approach to Design Case Adaptation*, Ph.D. Dissertation, Department of Architectural and Design Science, University of Sydney, Australia.

Gómez de Silva Garza, A. and Maher, M.L.: 1998, The Role of Non-Determinism in Computational Models of Creativity, in J.S. Gero and M.L. Maher (eds) *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, University of Sydney, Australia, pp. 47-55.

Gómez de Silva Garza, A. and Maher, M.L.: 1999, An Evolutionary Approach to Case Adaptation, in K.D. Althoff, R. Bergmann, and L.K. Branting (eds), *Case-Based Reasoning Research and Applications: Proceedings of the Third International Conference on Case-Based Reasoning ICCBR-99*, Springer-Verlag, Berlin, Germany, pp. 162-172.

Gómez de Silva Garza, A. and Maher, M.L.: 2000, A Process Model for Evolutionary Design Case Adaptation, in J. Gero (ed), *Artificial Intelligence in Design 2000*, 393-412, Kluwer Academic Publishers, Worcester, Massachusetts.

Graf, J. and Banzhof, W.: 1995, Interactive Evolution in Civil Engineering, in J.S. Gero, M.L. Maher, and F. Sudweeks (eds), *Third International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, Australia, pp. 303-316.

Hildebrand, G.: 1991, *The Wright Space: Pattern & Meaning in Frank Lloyd Wright's Houses*, University of Washington Press, Seattle.

Jo, J.: 1998, Interactive Evolutionary Design System, in J.S. Gero and M.L. Maher (eds), *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, Australia, pp. 215-223.

Kane, C. and Schoenauer, M.: 1996, Topological Optimum Design Using Genetic Algorithms, *Control and Cybernetics*, Vol. 25, No. 5.

Kolodner, J. L.: 1993, *Case-Based Reasoning*, Morgan Kaufmann, San Mateo, California.

Koning, H. and Eizenberg, J.: 1981, The Language of the Prairie: Frank Lloyd Wright's Prairie Houses, *Environment and Planning B*, Vol. 8, pp. 295-323.

Leake, D.B.: 1996, *Case-Based Reasoning: Experiences, Lessons, & Future Directions*, AAAI Press & The MIT Press, Menlo Park, California & Cambridge, Massachusetts.

Lenart, M. and Pasztor, A.: 1992, Creative Building Design by Using Genetic Algorithms, in J.S. Gero and F. Sudweeks (eds), *Second International Round-Table Conference on Computational Models of Creative Design*, Department of Architectural and Design Science, University of Sydney, Australia, pp. 129-144.

Louis, S.J.: 1992, Genetic Algorithms As A Computational Model of Creative Design, in J.S. Gero and F. Sudweeks (eds), *Second International Round-Table Conference on Computational Models of Creative Design*, Department of Architectural and Design Science, University of Sydney, Australia, pp. 243-273.

Louis, S.J., McGraw, G. and Wycoff, R.: 1993, Case-Based Reasoning Assisted Explanation of Genetic Algorithm Results, *Journal of Experimental and Theoretical Artificial Intelligence*, **5**:21-37.

Maher, M.L., Boulanger, S., Poon, J., and Gómez de Silva Garza, A.: 1995, Assessing Computational Methods with A Framework for Creative Design Processes, in J.S. Gero, M.L. Maher, and F. Sudweeks (eds), *Third International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, Australia, pp. 233-265.

Maher, M.L. and Wu, P.X.: 1998, Creativity Through Coevolutionary Design, in J.S. Gero and M.L. Maher (eds), *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, Australia, pp. 227-241.

Mitchell, M.: 1998, *An Introduction to Genetic Algorithms (Complex Adaptive Systems Series)*, MIT Press, Cambridge, Massachusetts.

Parmee, I. and Bonham, C.: 1998, Supporting Innovative and Creative Design Using Interactive Designer/Evolutionary Computing Strategies, in J.S. Gero and M.L. Maher (eds), *Fourth International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing and Cognition, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, Australia, pp. 187-213.

Prabhakar, S. and Goel, A.: 1992, Performance-Driven Creativity in Design: Constraint Discovery, Model Revision, and Case Composition, in J.S. Gero and F. Sudweeks (eds), *Second International Round-Table Conference on Computational Models of Creative Design*, Department of Architectural and Design Science, University of Sydney, Australia, pp. 101-127.

Qian, L. and Gero, J.S.: 1995, An Approach to Design Exploration Using Analogy, in J.S. Gero, M.L. Maher, and F. Sudweeks (eds), *Third International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, Australia, pp. 3-36.

Ramsey, C.L. and Grefenstette, J.J.: 1993, Case-Based Initialization of Genetic Algorithms, *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, California, pp. 84-91.

Rosenman, M.: 2000, Case-Based Evolutionary Design, *AIEDAM*, **14**(1):17-30.

Rossbach, S.: 1987, *Interior Design with Feng Shui*, Rider Books, London.

Wolverton, M. and Hayes-Roth, B.: 1995, Finding Analogues for Innovative Design, in J.S. Gero, M.L. Maher, and F. Sudweeks (eds), *Third International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, Australia, pp. 59-84.

Wright, F.L.: 1971, *The Natural House*, Pittman Publishing, London.

Zhao, F. and Maher, M.L.: 1992, Analogy and Mutation as Strategies for Creative Design, in J.S. Gero and F. Sudweeks (eds), *Second International Round-Table Conference on Computational Models of Creative Design*, Department of Architectural and Design Science, University of Sydney, Australia, pp. 53-100.